node.js 프로젝트 생성

2020-01-19

이승진

학습목표

javascript 프로젝트 생성

eslint 설정

javascript 반복문

지역변수 선언

목차

[1. eslint 2](#_Toc35768777)

[1) eslint란? 2](#_Toc35768778)

[2) eslint 설치 2](#_Toc35768779)

[2. bash 명령 3](#_Toc35768780)

[1) bash 명령 3](#_Toc35768781)

[2) 파일 경로명 (file path) 4](#_Toc35768782)

[3. node.js 프로젝트 5](#_Toc35768783)

[1) package.json 5](#_Toc35768784)

[2) eslint 설정 파일 생성 6](#_Toc35768785)

[3) 프로젝트 폴더 8](#_Toc35768786)

[4. 반복문 10](#_Toc35768787)

[1) chap02 폴더 생성 10](#_Toc35768788)

[2) chap02 폴더 아래 파일 생성 10](#_Toc35768789)

[3) for 문 11](#_Toc35768790)

[4) while 루프 13](#_Toc35768791)

[5) do while 루프 13](#_Toc35768792)

[6) break, continue 14](#_Toc35768793)

[5. 변수 선언 15](#_Toc35768794)

[1) 변수 선언 생략 15](#_Toc35768795)

[2) "use strict"; 16](#_Toc35768796)

[3) 상수 선언 const 17](#_Toc35768797)

[6. github 연동 18](#_Toc35768798)

[1) github repository 생성 18](#_Toc35768799)

[2) git 사용자 설정 19](#_Toc35768800)

[3) .gitignore 파일 생성 20](#_Toc35768801)

[4) local repository 생성 20](#_Toc35768802)

[5) 첫번째 commit 21](#_Toc35768803)

[6) github 서버에 push 22](#_Toc35768804)

[7) 수정한 소스 코드 등록하기. 22](#_Toc35768805)

[7. 내용 정리 24](#_Toc35768806)

[8. 과제 26](#_Toc35768807)

[1) 과제 프로젝트 생성 26](#_Toc35768808)

[2) 합계1 (sum1.js) 26](#_Toc35768809)

[3) 합계2 (sum2.js) 26](#_Toc35768810)

[4) 과제 프로젝트를 github에 등록하라. 26](#_Toc35768811)

# eslint

## eslint란?

오타 같은 실수에서 컴파일 에러가 발생해야 버그 찾기가 수월하다.

그런데 javascript는 인터프리터 방식의 언어라서 컴파일을 하지 않는다.

컴파일을 하지 않기 때문에, 컴파일 에러가 발생하지도 않는다.

소스코드에 오타가 있어도, 컴파일 에러가 발생하지 않고,

실행할 때 런타임 에러(runtime error)가 발생한다.

에러를 찾기 위해, 모든 기능을 다 실행해 보는 것은 힘들기 때문에,

컴파일 방식 언어보다 인터프리터 방식 언어는 버그 찾기가 힘들다.

이 불편함을 보완하는 도구가 eslint 이다.

eslint는 컴파일러처럼 소스코드를 분석해서 오타와 같은 문법 오류를 찾아준다.

바람직하지 않은 코딩 스타일도 지적해 준다.

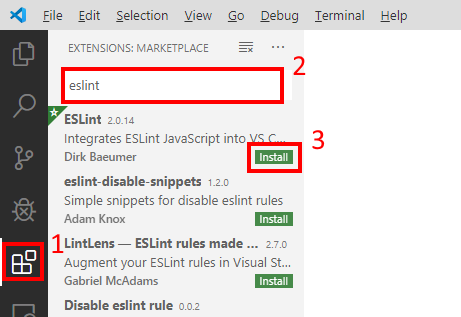
## eslint 설치

### eslint 설치 명령

|  |
| --- |
| npm install -g eslint |

|  |
| --- |
|  |

### visual studio code에 eslint 확장 설치



# bash 명령

Linux의 기본 쉘인 bash 기본 명령을 배워보자.

## bash 명령

디렉토리는 폴더와 같은 개념이다.

### 디렉토리 생성

|  |
| --- |
| **mkdir 디렉토리명** |

### 하위 디렉토리로 이동하기

|  |
| --- |
| **cd 디렉토리명** |

### 현재 디렉토리의 파일 목록 출력

|  |
| --- |
| **ls -al** |

### 상위 디렉토리로 이동하기

|  |
| --- |
| **cd ..** |

### 디렉토리 삭제하기

|  |
| --- |
| **rm -rf 디렉토리명** |

디렉토리와 그 아래 파일들이 모두 삭제된다.

### 파일 삭제하기

|  |
| --- |
| **rm 파일명** |

### 파일 복사하기 #1

|  |
| --- |
| **cp 소스\_파일 새\_파일** |

### 파일 복사하기 #2

|  |
| --- |
| **cp 소스\_파일 대상\_폴더** |

## 파일 경로명 (file path)

Windows 운영체제의 파일시스템에는 디스크 드라이브 개념이 있다. (예: C: D: E:)

Linux 에는 디스크 드라이브 개념이 없다.

Windows 에 설치한 Git bash 쉘에서, Windows 파일시스템의 디스크 드라이브를 다음과 같이 다룬다.

예를 들어 Windows 에서 파일 경로명이 다음과 같을 때

|  |
| --- |
| **C:**/PJ/node1/chap01/hello1.js |

위 경로명에 해당하는 **Git bash 경로명**은 다음과 같다.

|  |
| --- |
| **/c**/PJ/node1/chap01/hello1.js |

**Git bash**에서 C:/PJ/node1/ 디렉토리로 이동하는 명령은 다음과 같다.

|  |
| --- |
| cd /c/PJ/node1/ |

# node.js 프로젝트

## package.json

node.js 프로젝트의 설정 파일은 package.json 파일이다.

이 파일이 프로젝트 폴더에 있어야 한다.

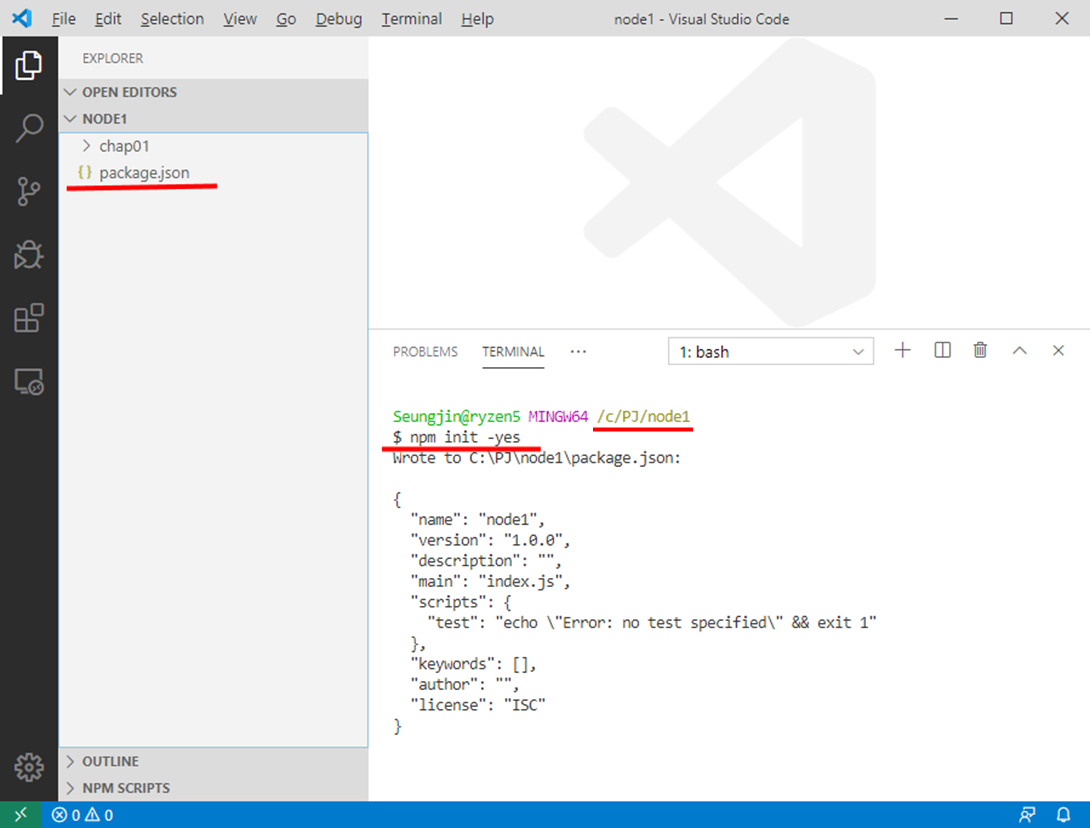
앞장에서 만든 node1 소스코드 폴더에 package.json 파일을 생성하자.

### node.js 프로젝트 설정 파일(package.json) 생성

주의: node1 프로젝트 폴더에서 아래 명령을 실행해야 한다.

|  |
| --- |
| npm init -yes |

node1 프로젝트 폴더에 package.json 파일이 생성된다.



## eslint 설정 파일 생성

프로젝트 폴더 에 eslint 설정 파일을 추가하자.

eslint 설정 파일은 .eslintrc 이다.

### 프로젝트에 eslint 설정 파일(.eslintrc) 추가 명령

주의: node1 프로젝트 폴더에서 아래 명령을 실행해야 한다.

|  |
| --- |
| eslint --init |

|  |
| --- |
|  |

위 아래 화살표 키를 눌러서 'To check syntax and find problems' 항목을 선택하고

엔터키 입력

check syntax : 문법 오류를 검사한다

find problems : 흔한 실수를 검사한다

enforce code style: 코드 스타일을 검사한다

|  |
| --- |
|  |

None of these 항목을 선택하고 엔터키 입력

|  |
| --- |
|  |

None of these 항목을 선택하고 엔터키 입력

|  |
| --- |
|  |

N을 입력하고 엔터키 입력

|  |
| --- |
|  |

스페이스 키를 눌러서 Browser 항목의 별표를 제거하고

Node 항목의 별표를 추가하고

엔터키 입력

Browser : 이 프로젝트의 코드가 웹브라우저에서 실행될 것이라면 선택

Node : 이 프로젝트의 코드가 node.js에서 실행될 것이라면 선택

|  |
| --- |
|  |

JSON 항목을 선택하고 엔터

(생성될 eslint 설정 파일의 포맷을 지정한다)

|  |
| --- |
|  |

Y를 입력하고 엔터키 입력

위 명령에 의해서, eslint의 설정 파일인 .eslintrc.json 파일 생성된다.

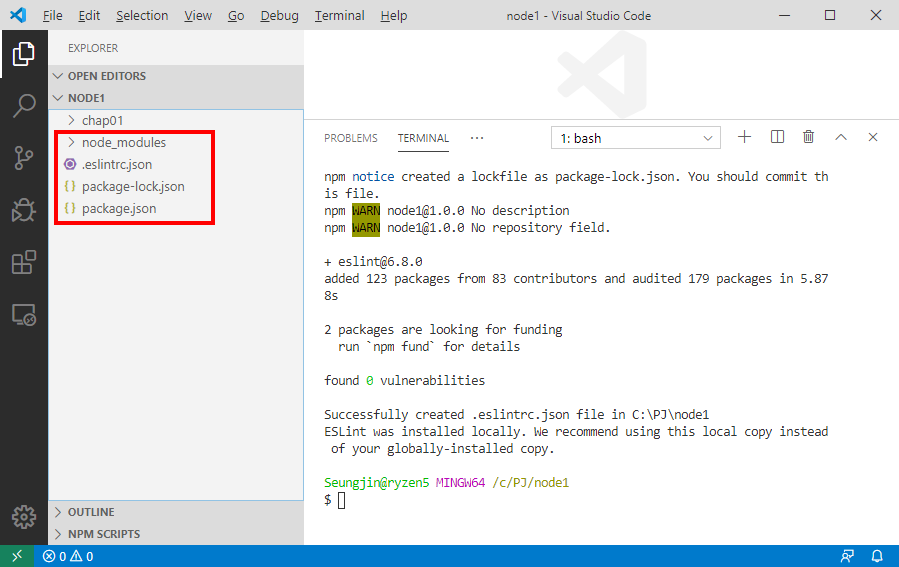
## 프로젝트 폴더

node1 프로젝트 폴더의 내용

- node\_modules 폴더

- package.jon 파일

- .eslintrc.json 파일



### package.json 파일의 내용

node.js 프로젝트 설정 파일

|  |
| --- |
| {    "name": "node1",    "version": "1.0.0",    "description": "",    "main": "index.js",    "scripts": {      "test": "echo \"Error: no test specified\" && exit 1"    },    "keywords": [],    "author": "",    "license": "ISC",    "devDependencies": {      "eslint": "^6.8.0"    }  } |

|  |  |
| --- | --- |
| 항목 | 설명 |
| name | 프로젝트 이름 |
| version | 프로젝트 버전 |
| description | 프로젝트 설명 |
| main | 프로젝트의 실행이 시작될 첫 파일 |
| scripts | 프로젝트를 빌드, 테스트, 실행하기 위한 명령 |
| keywords | 프로젝트의 검색 키워드 |
| author | 작성자 |
| license | 라이센스 정보 |
| devDependencies | 프로젝트에 필요한 라이브러리들 목록 |

### node\_modules 폴더

프로젝트에 필요한 라이브러리들이 자동으로 저장되는 폴더

### .eslintrc.json 파일

eslint 설정 파일

### package-lock.json

프로젝트에 필요한 라이브러리들의 정확한 버전이 자동으로 등록되는 파일.

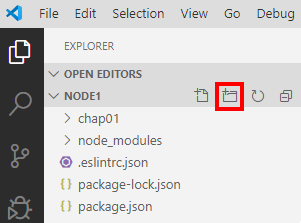
이 파일은 자동으로 생성된다.

지금은 이 파일을 무시하자.

# 반복문

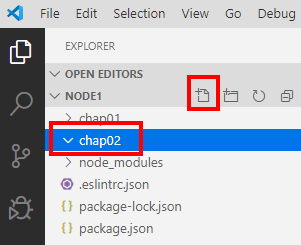
## chap02 폴더 생성

node1 프로젝트 폴더 아래에 chap02 폴더를 만들자.



node1 폴더를 클릭하고, 폴더 생성 버튼을 클릭하자.

## chap02 폴더 아래 파일 생성



chap02 폴더를 클릭하고, 파일 생성 버튼을 클릭하자.

## for 문

### for1.js

|  |
| --- |
| for (let i = 0; i < 10; ++i)      console.log(i); |

지역 변수 선언 키워드가 let 임에 주의하자.

javascript 언어는 변수를 선언할 때, 변수의 타입을 지정하지 않는다.

**int i = 0;** 문법 오류

**let i = 0;** Ok.

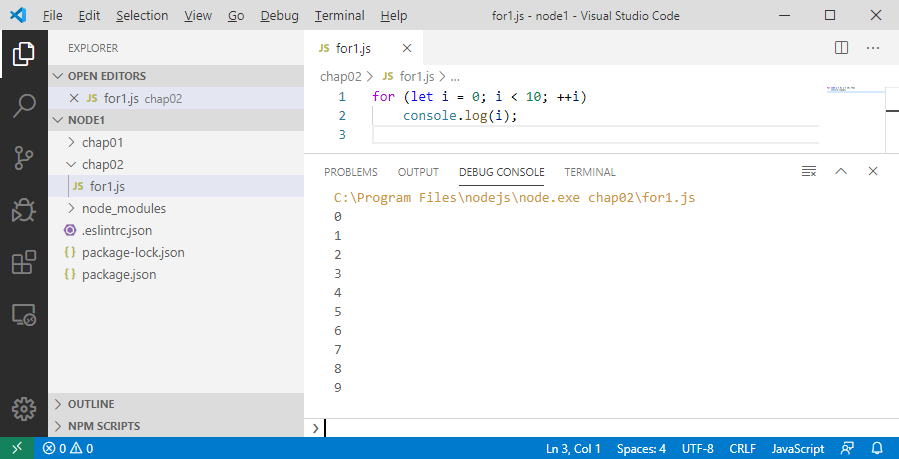
**String s = "hello";** 문법 오류

**let s = "hello";** Ok

let을 제외하면 for 문의 문법은 java와 동일하다.

### 실행

현재 파일 실행 단축키: Ctrl+F5



### for2.js

|  |
| --- |
| let s = "hello world";  for (let i = 0; i < s.length; ++i) {      let ch = s[i];      console.log(ch);  } |

let s = "hello world";

지역변수 s에 "hello world" 문자열이 대입된다.

문자열은 객체이다.

s.length

문자열 객체의 length 속성의 값은 문자열의 길이이다.

let ch = s[i];

s 문자열의 인덱스 i 위치 문자를, 지역변수 ch 에 대입한다.

실행

|  |
| --- |
|  |

### 참고 : For2.java

이것은 java 코드이다. (java 코드 실행은 이클립스에서...)

|  |
| --- |
| public class For2 {  public static void main(String[] args) {  String s = "hello world";  for (int i = 0; i < s.length(); ++i) {  char ch = s.charAt(i);  System.out.println(ch);  }  }  } |

for2.js 코드와 For2.java 코드를 비교하자.

Java 는 main 메소드에서 실행이 시작된다.

Java String 클래스의 length() 메소드의 리턴값은, 문자열의 길이이다.

Java String 클래스의 charAt(int index) 메소드의 리턴값은, 인덱스 i 위치 문자이다.

## while 루프

### while1.js

|  |
| --- |
| let i = 0;  while (i < 10) {      console.log(i);      ++i;  } |

while 문의 문법은 java와 동일하다.

실행

|  |
| --- |
|  |

## do while 루프

### do1.js

|  |
| --- |
| let i = 0;  do {      console.log(i);      ++i;  } while (i < 10); |

do while 문의 문법은 java와 동일하다.

실행

|  |
| --- |
|  |

## break, continue

반복문에서 break, continue 는 java 문법과 동일하다.

### loop1.js

|  |
| --- |
| let i = 0;  for (;;) {      ++i;      if (i % 2 == 1) continue;      console.log(i);      if (i >= 20) break;  } |

실행

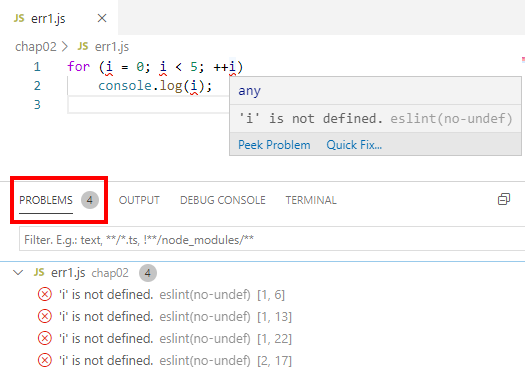
|  |
| --- |
|  |

# 변수 선언

## 변수 선언 생략

### err1.js

|  |
| --- |
| for (i = 0; i < 5; ++i)      console.log(i); |



위의 visual studio code 화면에서 변수 i 에 빨간색 밑줄이 표시되었다.

빨간색 밑줄 위에 마우스 포인터를 위치하고 기다리면,

위 화면과 같은 에러 메시지가 표시된다.

'i' is not defined.

변수 i 가 정의되지 않았다는 에러 메시지이다.

그런데, 단축키 Ctrl+F5를 눌러서 실행해 보면, 에러 없이 실행이 잘 된다.

|  |
| --- |
|  |

### 변수 선언을 생략해도 된다.

위 err1.js 에서 변수 i 를 선언하지 않고, 그냥 i = 0; 를 실행했다.

이것은 문법 오류가 아니다.

javascript 언어 문법에서는, 변수 선언을 생략해도 된다. 필수가 아니다.

i = 0; 문장을 처음 실행할 때, 변수 i 가 자동으로 생성되기 때문이다.

즉 javascript 문법에서는, 변수를 미리 선언하지 않아도, 어떤 변수에 값이 처음 대입될 때,

그 변수가 자동으로 생성된다.

### eslint 경고 메시지

위 화면에서, 변수 i 가 정의되지 않았다는 에러 메시지는,

javascript 문법 오류가 아니고, eslint 도구의 경고 메시지이다.

코딩 스타일이 바람직하지 않다는 경고 메시지이다.

변수를 미리 선언하지 않고 사용하는 것은 바람직하지 않기 때문이다.

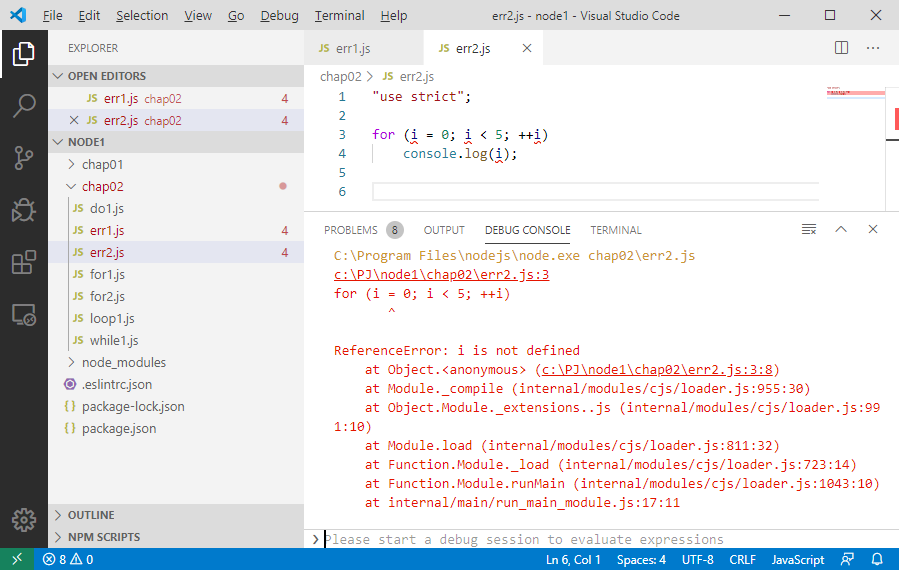
## "use strict";

javascript 소스 코드 파일 선두에 **"use strict";** 선언이 있으면,

이 소스 파일에서는 변수를 미리 선언하지 않고 사용할 경우에 문법 오류가 발생한다.

### err2.js

|  |
| --- |
| "use strict";  for (i = 0; i < 5; ++i)      console.log(i); |

  
단축키 Ctrl+F5 를 눌러서 실행하면,

위 화면에서처럼 문법 오류 메시지가 출력되고, 실행되지 않는다.

## 상수 선언 const

### const1.js

|  |
| --- |
| let s1 = "hello world";  const s2 = "hello world";  console.log(s1);  console.log(s2); |

const 키워드를 사용하여 상수를 선언한다.

s1 = "안녕하세요";

지역변수 s1의 값은 변경할 수 있다.

s2 = "안녕하세요";

const 변수의 값은 변경할 수 없으므로, 에러가 발생한다.

# github 연동

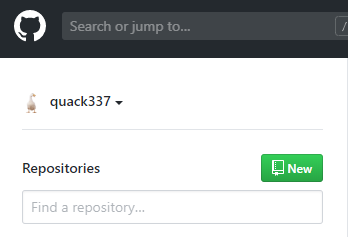
프로젝트 소스 코드를 github 서버에 저장하자.

## github repository 생성

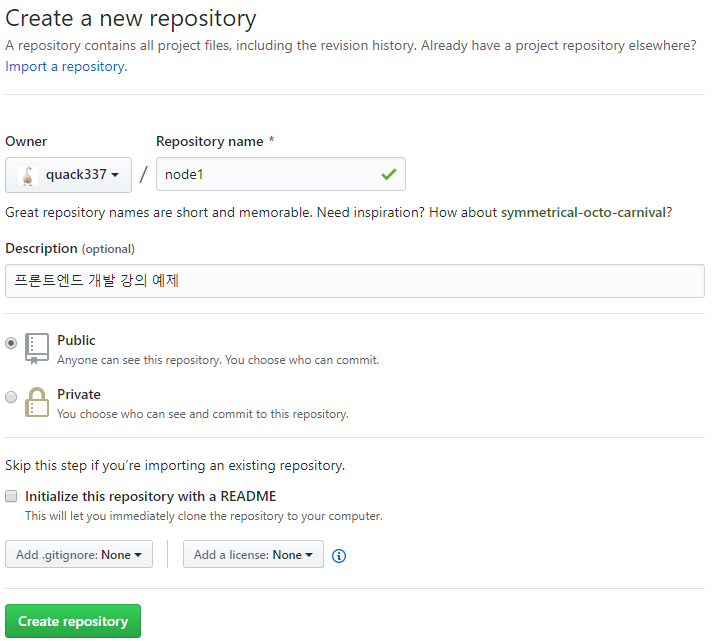
(1) 먼저 github에 회원가입하자.

gmail 주소로 회원가입하는 것이 좋다.

(2) 프로젝트를 저장할 repository를 만들자.



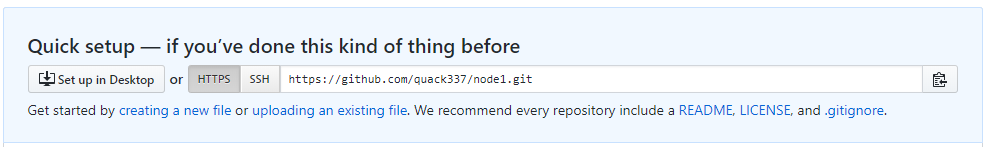
New 버튼 클릭



적당한 repository name을 입력하자. (프로젝트 폴더 이름을 입력하는 것이 무난함)

description에 간단한 설명을 입력하자.

create repository 버튼을 클릭하면 다음과 같은 화면이 나타난다.



위 화면의 URL이 방금 생성한 repository의 URL 이다.



위 두 명령을 나중에 실행해야 하므로, 잘 기록해 두자.

## git 사용자 설정

PC에서 git 사용자 정보를 설정해야 한다.

명령 프롬프트나 터미널에서 다음 명령을 실행하자.

|  |
| --- |
| git config --global user.name "사용자이름" |

혹시 한글이 깨질 수도 있으므로, 영어 이름을 입력하자.

|  |
| --- |
| git congig --global user.email 이메일주소 |

github 가입할 때 사용한 이메일 주소를 입력하자.

gmail 주소를 추천함.

|  |
| --- |
|  |

## .gitignore 파일 생성

프로젝트 폴더에 .gitignore 파일을 생성하자.

### .gitignore

|  |
| --- |
| node\_modules |

.gitignore 파일의 내용은, 버전 관리를 하지 않을 예외 파일이나 폴더 목록이다.

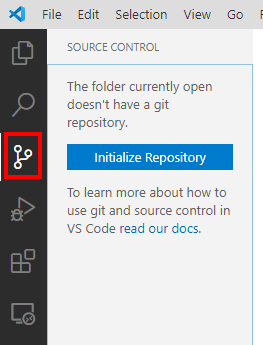
node\_modules 폴더에는 다운로드된 라이브러리 소스코드들이 저장된다.

이 소스코드들은 내가 작성한 코드가 아니고, 언제든 다시 다운로드 받을 수 있으므로,

버전관리 할 필요 없다.

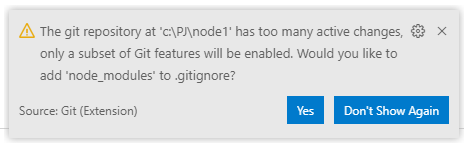
그리고 이 폴더의 용량이 너무 크기 때문에, 버전관리를 하면 무척 느려진다.

## local repository 생성



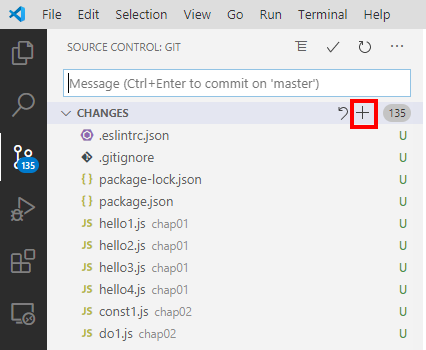
프로젝트 폴더가 열려있는 상태에서, visual studio code 화면 왼쪽의 툴바 버튼을 클릭하고

Initialize Repository 버튼을 클릭한다.



만약 화면 오른쪽 아래에 위와 같은 메시지가 보이면 Yes를 클릭하자.

## 첫번째 commit



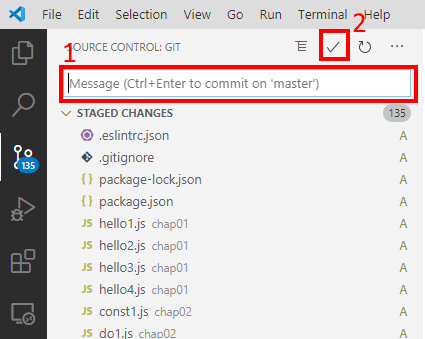
수정된 파일을 repository에 등록하기 위한 commit 화면이다.

먼저 repository에 등록할 파일들을 선택해야 한다.

위 화면에서 + 버튼을 클릭하면, 수정된 모든 파일이 등록 대상으로 선택된다. (stage all changes)

이 버튼을 클릭하면,

등록 대상으로 선택된 파일 목록 오른쪽의 문자가 U에서 A로 바뀐다.



최초 버전의 메시지를 입력하라는 화면이다.

텍스트 박스에 initial commit 이라고 입력하고

commit 버튼을 클릭하자.

위와 같이 하면, 수정된 소스 코드들이 local reposiotry에 등록(commit) 된다.

local repository는 내 PC의 repository 이다.

## github 서버에 push

local repository에 등록된 내용들을, remote repository에 전송(push) 해야한다.

remote repository는 github 서버에서 생성한 repository 이다.

먼저 local repository에 remote repository 주소를 등록해야 한다.

터미널이나 명령 프롬프트 창에서 다음 명령을 실행하자.

반드시 프로젝트 폴더에서 명령을 실행해야 한다.

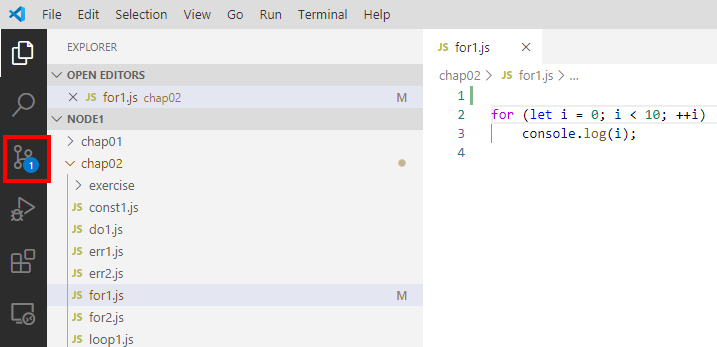
|  |
| --- |
| git remote add origin https://github.com/quack337/node1.git  git push -u origin master |

위 두 명령은 github에서 repository를 생성한 다음 화면에 보였던 명령이다.

첫째 줄이, local repository에 remote repository 주소를 등록하는 명령이다.

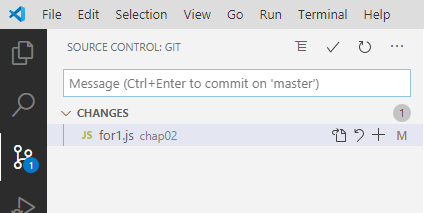
둘째 줄은, local repository에 등록된 내용들을, remote repository에 전송(push) 하는 명령이다.

## 수정한 소스 코드 등록하기.



소스 코드를 수정하면, 화면 왼쪽의 "source control" 버튼에 수정된 파일의 수가 나타난다.

이 "source control" 버튼을 클릭하자.



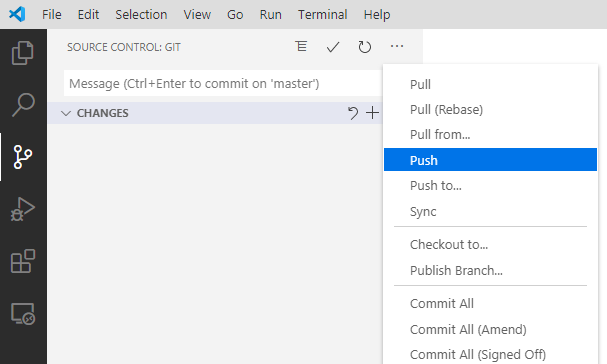
수정된 파일 목록이 나타난다.

수정된 파일 옆의 + 버튼을 클릭하면, 그 파일의 수정 사항이 등록 대상이 된다.

CHANGES 옆의 + 버튼을 클릭하면, 수정 사항 전체가 등록 대상이 된다.

텍스트 박스에 간단한 설명을 입력한 후, 위쪽의 commit 버튼을 클릭하여,

수정 사항을 local repository에 등록하자.



화면 위의 ... 버튼을 클릭하고, Push 메뉴를 클릭하면,

local repository에 등록된 내용이 remote repository에 전송(push) 된다.

# 내용 정리

### 디렉토리 생성

mkdir 디렉토리명

### 하위 디렉토리로 이동하기

cd 디렉토리명

### 현재 디렉토리의 파일 목록 출력

ls -al

### 상위 디렉토리로 이동하기

cd ..

### 디렉토리 삭제하기

rm -rf 디렉토리명

### 파일 삭제하기

rm 파일명

### 파일 복사하기 #1

cp 소스\_파일 새\_파일

### 파일 복사하기 #2

cp 소스\_파일 대상\_폴더

### Git bash에서 D:/pj 디렉토리로 이동하는 명령

cd /d/pj

### javascript 프로젝트 설정 파일

package.json

### javascript 프로젝트 설정 파일 생성 명령

npm init

### eslint

eslint는 컴파일러처럼 소스코드를 분석해서 오타와 같은 문법 오류를 찾아준다.

바람직하지 않는 코딩 스타일도 지적해 준다.

### eslint 설정 파일

.eslintrc.json

### eslint 설정 파일 생성 명령

eslint --init

### node\_modules 폴더

프로젝트에 필요한 라이브러리들이 자동으로 저장되는 폴더

### 변수 선언

javascript 문법에서는, 변수를 미리 선언하지 않고 사용해도 된다.

### "use strict";

javascript 소스 코드 파일 선두에 **"use strict";** 선언이 있으면, 변수 선언이 필수이다.

즉 변수를 미리 선언하고 사용해야 한다.

### 문자열.length

문자열의 길이

### 문자열[i]

문자열에서 인덱스 i 위치 문자

### const

const 키워드를 사용하여 상수를 선언한다.

# 과제

## 과제 프로젝트 생성

hw 이름의 node.js 프로젝트를 생성하라.

프로젝트에 elint 설정 파일을 생성하라.

## 합계1 (sum1.js)

1 이상 55 이하의 모든 정수를 더해서 화면에 출력하는 코드를 반복문으로 구현하시오.

### 답

|  |
| --- |
| let sum = 0;  for (let i = 1; i <= 55; ++i)    sum += i;  console.log(sum); |

## 합계2 (sum2.js)

1 이상 55 이하의 모든 3의 배수를 더해서 화면에 출력하는 코드를 반복문으로 구현하시오.

### 답

|  |
| --- |
| let sum = 0;  for (let i = 3; i <= 55; i += 3)    sum += i;  console.log(sum); |

javascript 언어의 연산자들은 java 언어의 그것과 거의 같다.

아래 URL을 참고하라.

<https://developer.mozilla.org/ko/docs/Web/JavaScript/Guide/Expressions_and_Operators#%EC%82%B0%EC%88%A0_%EC%97%B0%EC%82%B0%EC%9E%90>

## 과제 프로젝트를 github에 등록하라.

repository 이름은 임으로 정하고,

아래 링크를 열어서, 자신의 gmail 주소와, 과제 repository URL을 입력하시오.

월요일 수업

<https://docs.google.com/spreadsheets/d/1LNjpzKega6XxNncfLQ-I6RtAlt_mN4Jt8DbP1mgXYK4/edit?usp=sharing>

수요일 수업

<https://docs.google.com/spreadsheets/d/1_5t4GbE0xPPBIwTVW4hL4Wy-6d1I5jTpq_xsrNeJEX8/edit?usp=sharing>

과제 검사는 교수가 직접 github에 들어가서 할 것이기 때문에, 따로 제출하지 않아도 됨.

과제 repository에 과제가 아닌 다른 연습 소스코드가 더 있어도 됨. 과제 점수와 무관함.